Global Descent Replaces Gradient Descent to Avoid Local Minima Problem in Learning with Artificial Neural Networks

Bedri C. Cetin Joel W. Burdick and Jacob Barhen
Department of Electrical Engineering, Mechanical Engineering and
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91125

Abstract— **One of the fundamental limitations of artificial neural network learning by gradient descent is the susceptibility to local minima during training. This paper presents a new approach to learning, in which the gradient descent rule in the Backpropagation learning algorithm is replaced with a novel "Global Descent" formalism. This methodology is based on a recently developed global optimization scheme, acronymed TRUST, which formulates optimization in terms of the flow of a special deterministic dynamical system. We test the ability of the new dynamical system to overcome local minima with common benchmark examples and a pattern recognition example. The results demonstrate that the new method does indeed escape encountered local minima, and thus finds the global minimum solution to the specific problems.**

I. Introduction

Neural networks with massively parallel processing units provide an effective approach for a broad spectrum of applications – pattern mapping, pattern completion and pattern classification. The most influential development in this area was the invention of the Backpropagation algorithm [1], which is a systematic method for training multilayer artificial neural networks.

Despite its popularity, Backpropagation learning by gradient descent has two major drawbacks — the slow convergence time and the presence of local minima. First, there is no guarantee that the network can be trained in a reasonable amount of time because the convergence process may be exceedingly long. Second, there is no assurance that the network will train to the best configuration possible, since a local minimum can trap the training algorithm in an inferior solution, whereas ideally a global minimum is desired to solve the problem. In this paper we primarily discuss the latter problem and propose a novel methodology to overcome it. Emphasis will be given on those function learning tasks, where the network maps a set of input patterns to a set of output patterns.

Statistical training methods [2, 3, 4] have previously been proposed to alleviate the local minima problem. These methods introduce noise to connection weights during training, but suffer from extreme slow convergence due to their probabilistic nature. Here, we propose a new deterministic dynamical system, "Global Descent," which consists of a single vector differential equation. The system has recently been introduced [5] for general optimization problems and has been shown to be very effective in globally optimizing energy or cost functions due to its global descent property. In the context of artificial neural networks, Global Descent provides a simple extension to the Backpropagation algorithm by replacing the gradient descent method during training.

The new formalism has been tested for common benchmarks, like the XOR and parity functions, and also for a pattern recognition example using 60 patterns. The results demonstrate that Backpropagation associated with Global Descent escapes encountered local minima and in practice almost always converges to the globally minimal solution.

Section 2 reviews Backpropagation and the emergence of the local minima associated with gradient descent. Section 3 presents the Global Descent algorithm and discusses its behavior together with convergence properties. Section 4 provides the results of the benchmark simulations and, finally, section 5 summarizes our conclusions.

II. Local Minima Problem associated with Backpropagation

Backpropagation is a learning algorithm that gives a prescription for changing the connection weights of a feedforward network to learn a training set of input-output pairs without any prior knowledge of the mathematical function that maps them. More specifically, it uses gradi-

0-7803-0999-5/93/$03.00 ©1993 IEEE

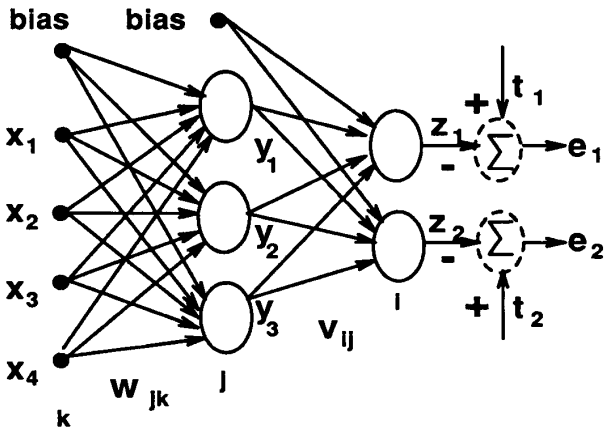

Figure 1: A two layer feed-forward network, showing the notation.

ent descent to adjust the weights following the local slope of the error surface towards a minimum. For the sake of completeness and clarity first we briefly review the Backpropagation algorithm below.

Without loss of generality, we assume a two-layered feed-forward network with one hidden layer, as shown in Figure 1. Let $\vec{x}^\mu$ and $\vec{t}^\mu$ $\mu = 1, 2, \ldots, p$ be the input and target output patterns respectively.

Given pattern μ, hidden unit j receives a net input h_j^μ and produces output y_j^μ,

$$h_j^\mu = \sum_k w_{jk}\, x_k^\mu \quad ; \quad y_j^\mu = s(h_j^\mu) \tag{1}$$

where, $s(.)$ is a sigmoidal function of the form $1/(1+exp(-(.)))$. Output unit i receives g_i^μ and produces the final output z_i^μ,

$$g_i^\mu = \sum_j v_{ij}\, y_j^\mu \quad ; \quad z_i^\mu = s(g_i^\mu)\,. \tag{2}$$

Bias to the units are not explicitly formulated but can be considered as an extra input clamped to $+1$ and connected to all units in the network.

As an error energy measure, or cost function, we choose the squares of the differences between the actual and target output values summed over the output units and all pairs of input/output patterns

$$E[\vec{w},\vec{v}] = \frac{1}{2}\sum_\mu^p \sum_i^I (t_i^\mu - z_i^\mu)^2 \;=\; \frac{1}{2}\sum_\mu^p \sum_i^I (e_i^\mu)^2 \tag{3}$$

where, $\vec{w}$ and $\vec{v}$ are the row-concatenated vector representation of the weight matrices $\mathcal{W}$ and $\mathcal{V}$ respectively. To train the network, the weights of each unit are adjusted to minimize the above energy function, thereby reducing the error between the actual and target outputs. In the Backpropagation algorithm, this is accomplished by using gradient descent, which changes the weights in proportion to the negative energy gradient. Thus, it is hoped to find a global minimum to (3), which will correspond to the optimal weights that solve the specific mapping problem.

To apply gradient descent, we first calculate the energy gradient for the hidden-to-output connections

$$\frac{\partial E[\vec{w},\vec{v}]}{\partial v_{ij}} = -\sum_\mu^p e_i^\mu\; s'(g_i^\mu)\; y_j^\mu \tag{4}$$

and then for the input-to-hidden connections

$$\frac{\partial E[\vec{w},\vec{v}]}{\partial w_{jk}} = -\sum_\mu^p \sum_i^I e_i^\mu\; s'(g_i^\mu)\; v_{ij}\; s'(h_j^\mu)\; x_k^\mu\,. \tag{5}$$

Finally, we apply gradient descent to equations (4) and (5) to get the weight dynamics

$$\dot{v}_{ij} = -\eta\,\frac{\partial E[\vec{w},\vec{v}]}{\partial v_{ij}} \quad ; \quad \dot{w}_{jk} = -\eta\,\frac{\partial E[\vec{w},\vec{v}]}{\partial w_{jk}} \tag{6}$$

where η is the learning rate.

Thus, training of the network consists mainly of weight adjustments performed according to (6). Clearly, gradient descent is a dynamical system whose stable equilibrium point only locally minimizes the error energy function. This works well with simple convex error surfaces, which have a unique minimum, but it often leads to nonoptimal and unacceptable solutions with the highly convoluted nonconvex surfaces encountered in practical problems. Once the algorithm gets trapped in a local minimum, application of more training iterations fails to improve learning. Ideally, only the global minimum of the error energy in equation (3) satisfies the convergence of the output patterns to the desired ones.

In the next section, we propose a novel dynamical system, called "Global Descent," which can be substituted for gradient descent in the Backpropagation algorithm to overcome the aforementioned limitations.

III. Global Descent Formalism

The "Global Descent" approach outlined below is based on a novel deterministic methodology for unconstrained global function optimization. This method, acronymed 'TRUST' [5], has been tested on standard benchmark functions and has been shown to be very efficient in locating the global minimum of multi-dimensional functions

with substantially faster rate of convergence than any competing global optimization technique [5]. The algorithm has already been employed, with encouraging results, to robotics applications [6, 7]. Since the primary purpose of this paper is the adaptation of the TRUST formalism to Backpropagation learning (hence named Global Descent) in artificial neural networks, we present here the methodology by only briefly reviewing its dynamics. Reference [5] should be consulted for greater details.

Global Descent formulates global optimization as the solution to a system of deterministic differential equations, where $E[\vec{w}, \vec{v}]$ of (3) is the function to be optimized with the connection weights being the states of the system. Let $\vec{\varphi}$ denote $(\vec{w}, \vec{v})$ and have elements φ_{ab}. Global Descent is based on the following equation

$$\dot{\varphi}_{ab} = -\eta \frac{\partial E[\vec{\varphi}]}{\partial \varphi_{ab}} \frac{1}{1+\exp(E[\vec{\varphi}] - E[\vec{\varphi}^*] + \sigma)} + \eta\, k\, (\varphi_{ab} - \varphi^*_{ab})^{1/3} u(E[\vec{\varphi}] - E[\vec{\varphi}^*]) \qquad (7)$$

where, $\vec{\varphi}^*$ is a fixed value of $\vec{\varphi}$, which can be a local minimum or an initial weight state, $u(.)$ is the Heaviside step function, and σ is a shifting parameter (its influence is discussed in [5], in numerical applications we typically take $\sigma = 2$). The first term in the RHS of equation (7) is a "subenergy gradient," with $1/(1+\exp(E[\vec{\varphi}] - E[\vec{\varphi}^*] + \sigma))$ being the "gradient multiplier," while the second term is a "non-lipschitzian terminal repeller," whose properties will be discussed further below. The parameter $k > 0$ is referred to as the "power" of the repeller, whose magnitude can be determined using the analysis of [5]. In the simulations of Section 4, $k = 0.001$ gave good results.

The dynamics in (7) is achieved upon application of gradient descent to the "cost function"

$$C[\vec{\varphi}, \vec{\varphi}^*] = \log\left(\frac{1}{1+\exp(-(E[\vec{\varphi}] - E[\vec{\varphi}^*] + \sigma))}\right) - k\frac{3}{4}\sum_{ab}(\varphi_{ab} - \varphi^*_{ab})^{4/3} u(E[\vec{\varphi}] - E[\vec{\varphi}^*]). \qquad (8)$$

The first term in the RHS of equation (8) is a nonlinear, but monotonic transformation of $E[\vec{\varphi}]$, which preserves all of its properties relevant for optimization, i.e. it has the same critical points as $E[\vec{\varphi}]$ and the same relative ordering of the local and global minima. Additionally, it works like a filter by flattening only the portions of the energy surface $E[\vec{\varphi}]$ which lie above $E[\vec{\varphi}^*]$ and leaves it nearly unmodified elsewhere. The term $\sum_{ab}(\varphi_{ab} - \varphi^*_{ab})^{4/3}$ is referred to as the "repeller energy term," which creates a convex surface with a unique minimum located at $\vec{\varphi} = \vec{\varphi}^*$. In effect, as seen in Figure 2, $C[\vec{\varphi}, \vec{\varphi}^*]$ of (8) transforms

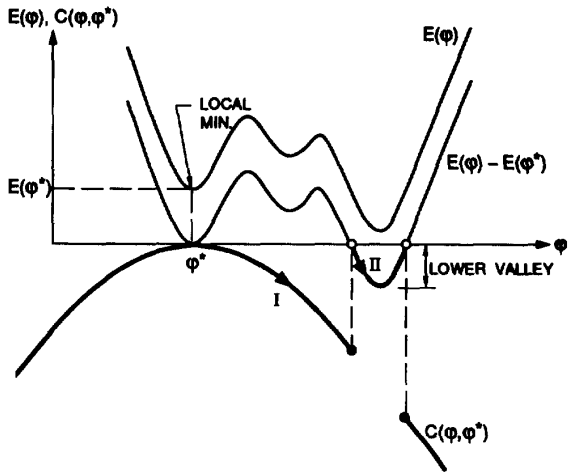

Figure 2: A one-dimensional schematical representation of the cost function $C[\varphi, \varphi^*]$ for an arbitrary error energy $E[\varphi]$.

the current local minimum of $E[\vec{\varphi}]$ into a unique maximum such that gradient descent can escape from it to a lower valley.

Thus, when the gradient descent in equation (6) is replaced by the Global Descent of (7), the Backpropagation algorithm escapes the encountered local minimum of the multidimensional functional $E[\vec{\varphi}]$ of (3), due to the following characteristics of the Global Descent system.

The dynamical system (7) autonomously switches between the following two phases:

Phase I This phase, which is effectively a tunneling phase, is characterized by $E[\vec{\varphi}] \geq E[\vec{\varphi}^*]$. Since for this condition the subenergy gradient magnitude is nearly zero in the vicinity of the local minimum $(\vec{\varphi}^*)$, the dynamical system (7) behaves approximately as:

$$\dot{\varphi}_{ab} \simeq \eta\, k\, (\varphi_{ab} - \varphi^*_{ab})^{1/3}$$

This system represents a terminal repeller, originally introduced by Zak [8], which has a repelling unstable equilibrium point at $\varphi_{ab} = \varphi^*_{ab}$, i.e. at the local minimum of the energy functional $E[\vec{\varphi}]$. Thus, due to this repeller, the dynamical system (7), when initialized with a small perturbation from $(\vec{\varphi}^*)$, will be repelled from the local minimum until it reaches a lower basin of attraction, where $E[\vec{\varphi}] < E[\vec{\varphi}^*]$. In

effect, this phase tunnels through portions of $E[\vec{\varphi}]$ where $E[\vec{\varphi}] \geq E[\vec{\varphi}^*]$.

Phase II This phase, which is a minimization phase, is characterized by $E[\vec{\varphi}] < E[\vec{\varphi}^*]$. The gradient multiplier term has approximately unit magnitude, and the repeller term is identically zero. Thus (7) behaves as:

$$\dot{\varphi_{ab}} \simeq -\eta \frac{\partial E[\vec{\varphi}]}{\partial \varphi_{ab}}.$$

Clearly this phase implements minimization via gradient descent.

In order to provide a clearer picture of the Backpropagation with Global Descent, we outline its implementation in terms of a step-by-step procedure. First let us assume the two-layered feed-forward network has K units, with M input-to-hidden weights and N hidden-to-output weights, and will be trained for p input-output patterns.

1. Calculating the activation of the units, defining the error energy and finding the energy gradients will be performed using equations (1) through (5) as in standard Backpropagation. However, training will be performed by using Global Descent dynamics in (7), which will be substituted for the former gradient descent law of equation (6).

2. To initiate the dynamics, we pick an arbitrary domain in the form of hyper-parallelpiped of dimension $M + N$, and choose $(\vec{\varphi}^*)$ as one corner of the domain. In effect, a repeller is placed at $(\vec{\varphi}^*)$ and the dynamical system in (7) is given initial conditions $(\vec{\varphi}^* + \vec{\epsilon}_{\varphi})$ where $\vec{\epsilon}_{\varphi}$ is a small perturbation which drives the system into the domain of interest.

3. If $E[\vec{\varphi}^* + \vec{\epsilon}_{\varphi}] < E[\vec{\varphi}^*]$, the system immediately enters a gradient descent phase (phase II above), which equilibrates at a local minimum $(\vec{\varphi}^{1*})$.

4. We then set $(\vec{\varphi}^*) = (\vec{\varphi}^{1*})$, and perturb $(\vec{\varphi})$ to $(\vec{\varphi}^{1*} + \vec{\epsilon}_{\varphi})$. Note that consistency in the flow direction is necessary.

5. Since $(\vec{\varphi}^{1*})$ is a local minimum, $E[\vec{\varphi}] \geq E[\vec{\varphi}^{1*}]$ holds in a neighborhood of $(\vec{\varphi}^{1*})$. Thus, the system enters the repelling phase (phase I above), and the repeller located at $(\vec{\varphi}^{1*})$ repels the system until it reaches a lower basin of attraction, where $E[\vec{\varphi}] < E[\vec{\varphi}^{1*}]$. In effect, this phase tunnels through all of the state space region with error energy values that lie above the last found lower local minimum. As the dynamical system enters the next basin, the algorithm automatically switches to gradient descent, leading to minimization of $E[\vec{\varphi}]$ in (3). By using the energy gradient flow the system will equilibrate at the next lower local minimum, $(\vec{\varphi}^{2*})$. We then set $(\vec{\varphi}^*) = (\vec{\varphi}^{2*})$ and repeat the process.

6. If $E[\vec{\varphi}^* + \vec{\epsilon}_{\varphi}] \geq E[\vec{\varphi}^*]$ when the training is initiated, (7) is initially in a tunneling phase. The tunneling will proceed to a lower basin, at which point it enters the minimization phase and follows the behavior discussed above.

7. The successive minimization and tunneling computational processes continue until a suitable stopping criterion is satisfied. If an exact solution for the weights exist such that for the specific problem input patterns can be accurately mapped to the target output patterns, then the global minimum at $E[\vec{\varphi}] = 0$ will set the stopping criteria. However if an exact solution does not exist, the global minimum associated with $E[\vec{\varphi}] > 0$ can be located by ocular inspection (it is detected as the lowest local minimum) of the energy $E[\vec{\varphi}]$ vs. the number of training epochs curve, which will indicate the optimal solution to the problem. Additionally the global minimum satisfies the criteria of a local minimum (i.e. energy gradients for all weights being close to zero).

As stated earlier, [5] provides further detailed description of the convergence properties of the algorithm. It is worth noting that training the network in the batch mode (i.e. weights are adjusted after presenting all input-output patterns) is required, since the system in (7) necessitates a unique energy surface $E[\vec{\varphi}]$ of (3) for all patterns.

Evidently, the structure of the dynamical system in (7) is highly parallel, allowing implementation in a form whose computational complexity is only weakly dependent on problem dimensionality (i.e. $M + N$ here).

IV. Examples and Applications

This section presents results of benchmark tests carried out for comparison of standard Backpropagation and Backpropagation associated with Global Descent. Our first two examples are problems that are often used for benchmarking a network [9]: the XOR and the parity problem. We also consider a pattern recognition example of 60 input-output patterns as an application. In all of them we demonstrate that for some initial weights, learning with the standard Backpropagation gets caught in a local minimum and either gives an inferior solution or even may be unable to solve the problem. Backpropagation with Global Descent on the other hand escapes the encountered local minima and converges to the global

Table 1: Input, target, output results and local minimum weight states for XOR function

pattern #	x_1	x_2	Target	Local	Global
1	0.0	0.0	0.0	0.056	0.030
2	0.0	1.0	1.0	0.961	0.963
3	1.0	0.0	1.0	0.493	0.964
4	1.0	1.0	0.0	0.498	0.031

Local Minimum Weights		
$w_{10} = 1.3643$	$w_{20} = 5.5906$	$v_{10} = 9.7378$
$w_{11} = -18.58$	$w_{21} = 19.143$	$v_{11} = -10.88$
$w_{12} = -5.4145$	$w_{22} = -7.6285$	$v_{12} = -9.7378$

minimum of the error energy function. For simplicity we will refer to the former as gradient descent and latter as Global Descent.

A. The XOR Problem

We begin with the exclusive-or problem since it is a classical problem requiring hidden units and since many other difficult problems involve an XOR as a subproblem. The truth table with the input and target patterns of the XOR function is shown in the first four columns of Table 1. One of the common problems in doing the XOR problem with a standard back-error paradigm is the presence of the local minima [10, 11].

We employed the smallest number of units in the network that can accomplish the XOR function: two units in the hidden layer, one in the output layer. The network required two external inputs, and bias units as demonstrated in Figure 1. The network with nine connection weights presented a 9-dimensional optimization problem. 50% of all simulations with random initial weights got stuck in a local minimum when using gradient descent. Figure 3 shows a comparison simulation of gradient descent and Global Descent for a specific random initial weight set. As the dashed line indicates, gradient descent got caught in a local minimum causing 12.74% error (i.e. 0.255/2.0) and thus evaluated only the first two entries in the training set correctly and failed for others as indicated in the fifth column of Table 1. Application of more training iterations in this case failed to get better convergence. Global Descent, on the other hand, as the solid line of Figure 3 shows, escaped the local minimum by tunneling through functionally higher values of the error function (seen as hill) and located the global minimum with only 0.11% error and solved the XOR problem as demonstrated in the sixth column of Table 1.

It is also worth noting that gradient descent fails to

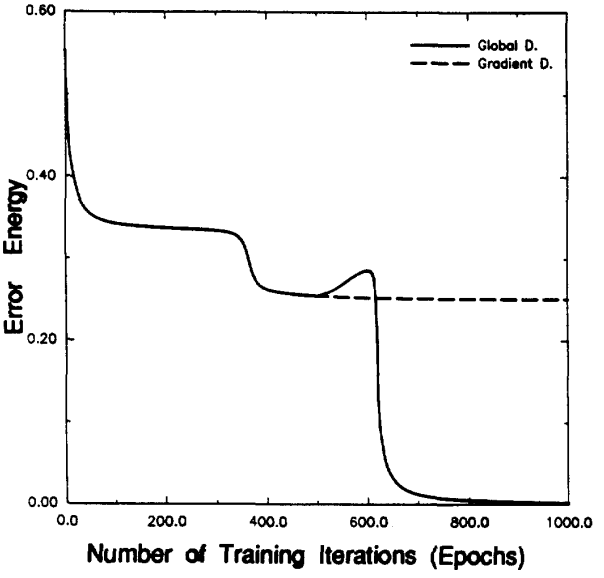

Figure 3: Global vs. Gradient Descent for XOR function

escape the critical point even after four million training iterations. Furthermore, to disprove the popular speculation that the XOR and similar functions do not have local minima but rather possess almost flat regions with extremely small slopes [12], we have calculated the eigenvalues of the Hessian matrix for the local weights shown in Table 1, and found that they all have positive values which proves the existence of true local minima.

B. The Parity Problem

The parity problem is essentially a generalization of the XOR problem to K inputs and has been extensively discussed by Minsky and Papert [13]. The single output unit is required to be on if an odd number of inputs are on, and off otherwise. It is often used for evaluating network performances and has been classified as a challenging problem, since the output changes whenever any single unit changes, i.e., the most similar input patterns correspond to different target patterns. This fact has been observed to cause local minima for certain initial weights.

The network topology has four inputs, four units in the hidden layer and and a single output unit. Thus training the network gives rise to a 25-dimensional optimization problem, (i.e., $E[\vec{\varphi}]$ in (3) contains 25 states as the connection weights). The training set has 16 input-target patterns. Figure 4 shows a performance comparison be-

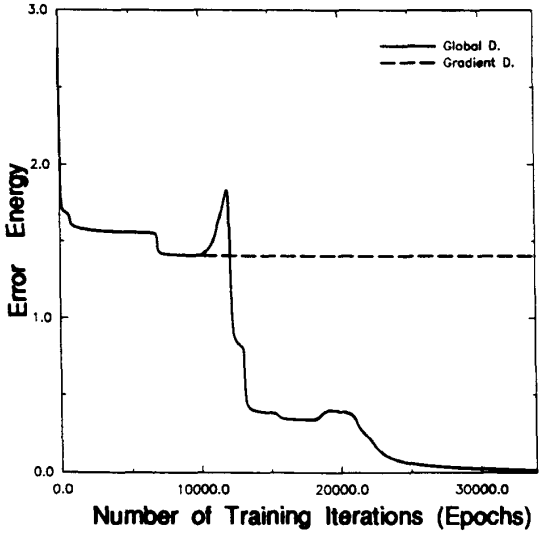

Figure 4: Global vs. Gradient Descent for Parity function

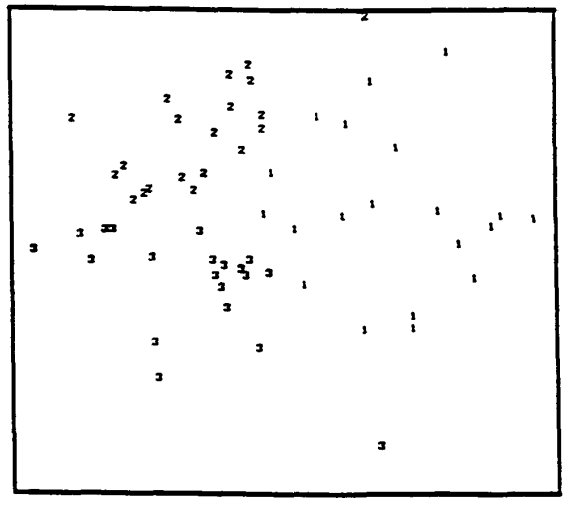

Figure 5: Training patterns for Pattern Recognition

tween the aforementioned methods for a specific random initial weight set. While gradient descent produced an unacceptable local solution with a 17.59% error, Global Descent after escaping through two consecutive local minima converged to the global solution with 0.22% error.

C. A Pattern Recognition Example

We also considered a simple pattern recognition example of 60 patterns [14]. Figure 5 shows a set of two-dimensional training patterns from three classes. This requires the design of a neural network recognizer with three output neurons, each of which shall be on if the coordinates of a sample of the corresponding class is presented, i.e. a "winner take all" network is necessary. Several simulations suggested that at least two units in the hidden layer were required to solve the problem. Figure 6 presents our results. Gradient descent trained the network to learn the patterns with 13.77% (i.e., 12.4/90) error due to a local minimum in the 15-dimensional error energy. Performing the same experiment with Global Descent solved the problem globally with a 0.32% total error for all 60-patterns.

It is important to note that in all the simulations above, we mainly concentrated on the local minima problem. The issue of rate of convergence was not the objective of this paper, therefore we used the simple Euler integra-

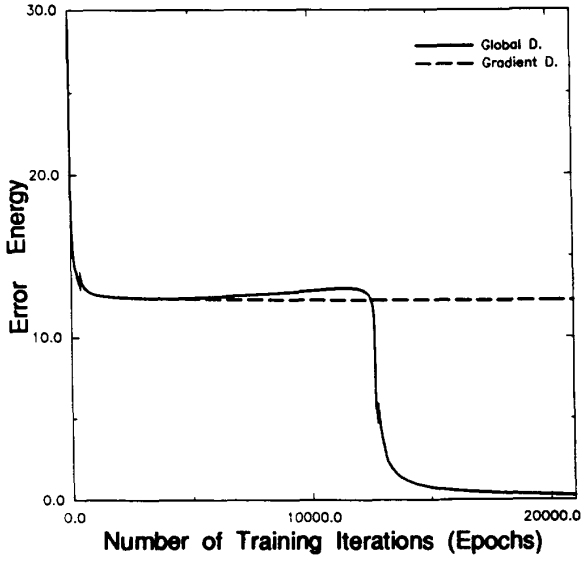

Figure 6: Global vs. Gradient Descent for Pattern Recognition

tion scheme which resulted in a relatively large number of iterations (i.e., epochs). Better integration schemes can speed up convergence.

V. Conclusion

This paper has introduced Global Descent in the context of artificial neural networks to improve learning. The methodology, which is based on a deterministic dynamical system, is proposed as a candidate for replacing the gradient descent formalism in the Backpropagation algorithm, in order to eliminate the local minima problem. Theoretically, convergence of the method to a global minimum is not formally guaranteed due to the constant perturbation direction vector $\vec{\epsilon}_{\varphi}$. However, in practice, due to its global descent property, the system dynamics escapes local minima valleys with help of the repeller effect, and flows into lower valleys of the error energy function using the information it gets from the gradient term. Indeed, previous studies [5, 6, 7] and the benchmark simulations of Backpropagation with Global Descent demonstrate that it can escape encountered local minima, and in almost all cases converges to the globally optimal solution of a specific problem.

References

[1] Rumelhart, D.E., Hinton, G.E., and Williams R.J., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, pp. 318-362, MIT Press, Cambridge, MA, 1986.

[2] Hinton, G.E., Sejnowski, T.J., "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing*, Vol. 1, pp 282-317, MIT Press, Cambridge, MA, 1986.

[3] Guillerm, T.J., Cotter, N.E., "A Diffusion Process for Global Optimization in Neural Networks," *Proc. IEEE INNS International Joint Conference on Neural Networks - Seattle*, Vol 1, pp. 335-340, 1991.

[4] Wasserman P.D., "Backpropagation and Cauchy Training: an Overview," in *Neural Computing: Theory and Practice*, pp. 87-92, VNR Press, New York, NY, 1989.

[5] Cetin, B.C., Barhen, J., Burdick, J.W., "Terminal Repeller Unconstrained Subenergy Tunneling (TRUST) for Fast Global Optimization," *Journal of Optimization Theory and Applications*, Vol. 77, April 1993, in press.

[6] Burdick J.W., Cetin, B.C., Barhen, J., "Efficient Global Redundant Configuration Resolution via Subenergy Tunneling and Terminal Repelling," *Proc. IEEE Inter. Conf. on Robotics and Automation*, Sacramento, CA, April 1991.

[7] Chen, I., Burdick J.W., "Finding Antipodal Point Grasps on Irregularly Shaped Objects," *Proc. IEEE Inter. Conf. on Robotics and Automation*, Nice, France, May 1992.

[8] Zak, M., "Terminal Attractors in Neural Networks," *Neural Networks*, Vol. 2, pp. 258-274, 1989.

[9] Hertz, J., Krogh, A., Palmer, R.G., "Backpropagation: Examples and Applications," in *Introduction to the Theory of Neural Computation*, pp. 130-141, Addison-Wesley, 1991.

[10] Dayhoff, J., "The Exclusive-Or: A classic problem," in *Neural Network Architectures: an Introduction*, pp. 76-79, VNR Press, New York, NY, 1990.

[11] Blum, E.K., "Approximation of Boolean Functions by Sigmoidal Networks: Part I: XOR and Other Two-Variable Functions," *Neural Computation*, Vol. 1, pp.532-540, 1989.

[12] McInerney, J.M., Haines, K.G., Biafore, S., Hecht-Nielsen, R., "Error Surfaces of Multi-Layer Networks Can Have Local Minima," *Technical Report # CS89-157*, UCSD, CA 1989.

[13] Minsky, M., Papert, S., *Perceptrons*, MIT press, Cambridge, MA, 1969.

[14] Atiya A., "Learning Algorithms for Neural Networks," *PhD Thesis*, Caltech, CA, 1991.